```
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFF               000         000 RRR         RRR RRR         RRR       TTT        LLL
FFFFFFFFFFFF      000         000 RRRRRRRRRRRR    RRRRRRRRRRRR          TTT        LLL
FFFFFFFFFFFF      000         000 RRRRRRRRRRRR    RRRRRRRRRRRR          TTT        LLL
FFFFFFFFFFFF      000         000 RRRRRRRRRRRR    RRRRRRRRRRRR          TTT        LLL
FFF               000         000 RRR    RRR      RRR    RRR            TTT        LLL
FFF               000         000 RRR    RRR      RRR    RRR            TTT        LLL
FFF               000         000 RRR    RRR      RRR    RRR            TTT        LLL
FFF               000         000 RRR        RRR  RRR        RRR        TTT        LLL
FFF               000         000 RRR        RRR  RRR        RRR        TTT        LLL
FFF               000         000 RRR        RRR  RRR        RRR        TTT        LLL
FFF                 000000000     RRR            RRR RRR            RRR   TTT        LLLLLLLLLLLLLLL
FFF                 000000000     RRR            RRR RRR            RRR   TTT        LLLLLLLLLLLLLLL
FFF                 000000000     RRR            RRR RRR            RRR   TTT        LLLLLLLLLLLLLLL
```

FILEID**FOROPEN

```
FFFFFFFFFF    000000    RRRRRRRR      000000    PPPPPPPP    EEEEEEEEEE  NN      NN
FFFFFFFFFF    000000    RRRRRRRR      000000    PPPPPPPP    EEEEEEEEEE  NN      NN
FF          00      00  RR      RR  00      00  PP      PP  EE          NN      NN
FF          00      00  RR      RR  00      00  PP      PP  EE          NN      NN
FF          00      00  RR      RR  00      00  PP      PP  EE          NNNN    NN
FF          00      00  RR      RR  00      00  PP      PP  EE          NNNN    NN
FFFFFFFF    00      00  RRRRRRRR    00      00  PPPPPPPP    EEEEEEEE    NN  NN  NN
FFFFFFFF    00      00  RRRRRRRR    00      00  PPPPPPPP    EEEEEEEE    NN  NN  NN
FF          00      00  RR   RR     00      00  PP          EE          NN    NNNN
FF          00      00  RR   RR     00      00  PP          EE          NN    NNNN
FF          00      00  RR    RR    00      00  PP          EE          NN      NN      ....
FF          00      00  RR    RR    00      00  PP          EE          NN      NN      ....
FF            000000    RR      RR    000000    PP          EEEEEEEEEE  NN      NN      ....
FF            000000    RR      RR    000000    PP          EEEEEEEEEE  NN      NN      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1        0001  0 MODULE FOR$OPEN (%TITLE 'FORTRAN OPEN'
;     2        0002  0                   IDENT = '1-065'          ! File: FOROPEN.B32  Edit: SBL1065
;     3        0003  0                   ) =
;     4        0004  1 BEGIN
;     5        0005  1 !
;     6        0006  1 !**********************************************************************************
;     7        0007  1 !*                                                                                *
;     8        0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                       *
;     9        0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                         *
;    10        0010  1 !*  ALL RIGHTS RESERVED.                                                           *
;    11        0011  1 !*                                                                                *
;    12        0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED         *
;    13        0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE        *
;    14        0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER        *
;    15        0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY        *
;    16        0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY        *
;    17        0017  1 !*  TRANSFERRED.                                                                   *
;    18        0018  1 !*                                                                                *
;    19        0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE        *
;    20        0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT        *
;    21        0021  1 !*  CORPORATION.                                                                   *
;    22        0022  1 !*                                                                                *
;    23        0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS        *
;    24        0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                        *
;    25        0025  1 !*                                                                                *
;    26        0026  1 !*                                                                                *
;    27        0027  1 !**********************************************************************************
;    28        0028  1
;    29        0029  1 !++
;    30        0030  1 ! FACILITY: FORTRAN Support Library - user callable
;    31        0031  1 !
;    32        0032  1 ! ABSTRACT:
;    33        0033  1 !
;    34        0034  1 !       This module opens a file on a specified logical unit
;    35        0035  1 !       (LUN) and allocates 3 control blocks for use by subsequent
;    36        0036  1 !       I/O statement calls for this LUN. The 3 control blocks
;    37        0037  1 !       are: Logical Unit Block (LUB), I/O statement Block (ISB),
;    38        0038  1 !       and an RMS Record Access Block (RAB).
;    39        0039  1 !
;    40        0040  1 ! ENVIRONMENT: User access mode; mixture of AST level or not.
;    41        0041  1 !
;    42        0042  1 ! AUTHOR:       Thomas N. Hastings, CREATION DATE: 6-Mar-77; Version 0
;    43        0043  1 !
;    44        0044  1 ! MODIFIED BY:
;    45        0045  1 !
;    46        0046  1 !       Thomas N. Hastings, 15-Mar-77: Version 0
;    47        0047  1 ! [Previous edit history removed.  SBL 5-Oct-1982]
;    48        0048  1 ! 1-062 - Move the BUILTIN ACTUALCOUNT into the routine that needs it, in
;    49        0049  1 !         anticipation of the next BLISS compiler, which will require it
;    50        0050  1 !         to be there.  While we are here, improve the source text layout.
;    51        0051  1 !         Note that this edit changes no code.  JBS 27-Aug-1980
;    52        0052  1 ! 1-063 - Add support for DEFAULTFILE keyword.  JAW 30-Jun-1981
;    53        0053  1 ! 1-064 - Allow DEFAULTFILE value to be ASCIZ.  JAW 30-Jun-1981
;    54        0054  1 ! 1-065 - Reflect separation of FOR$$ data structures from FOR$$.  SBL 5-Oct-1982
;    55        0055  1 !--
;    56        0056  1
```

```
;    58          0057  1 !
;    59          0058  1 ! PROLOGUE FILE:
;    60          0059  1 !
;    61          0060  1
;    62          0061  1 REQUIRE 'RTLIN:FORPROLOG';                      ! FORTRAN Declarations
;    63          0127  1
;    64          0128  1 !
;    65          0129  1 ! TABLE OF CONTENTS:
;    66          0130  1 !
;    67          0131  1
;    68          0132  1 FORWARD ROUTINE
;    69          0133  1     FOR$OPEN,                                   ! FORTRAN OPEN statement
;    70          0134  1     FOR$$OPECLO_ARG : NOVALUE,                  ! Get OPEN/CLOSE arguments
;    71          0135  1     OPEN_ON_CONNECTED : CALL_CCB;               ! open on a connected unit
;    72          0136  1
;    73          0137  1 !
;    74          0138  1 ! MACROS:
;    75          0139  1 !
;    76          0140  1 !       NONE
;    77          0141  1 !
;    78          0142  1 ! EQUATED SYMBOLS:
;    79          0143  1 !
;    80          0144  1 !       NONE
;    81          0145  1 !
;    82          0146  1 ! OWN STORAGE:
;    83          0147  1 !
;    84          0148  1 !       NONE
;    85          0149  1 !
;    86          0150  1 ! EXTERNAL REFERENCES:
;    87          0151  1 !
;    88          0152  1
;    89          0153  1 EXTERNAL ROUTINE
;    90          0154  1     FOR$$ERR_OPECLO,                            ! OPEN/CLOSE condition handler
;    91          0155  1     FOR$$OPEN_PROC : CALL_CCB NOVALUE,          ! Does the actual OPEN
;    92          0156  1     FOR$$SIGNAL_STO : NOVALUE,                  ! Convert small FORTRAN err #
;    93          0157  1                                                 ! to 32-bit VAX error # and SIGNAL_STOP
;    94          0158  1     FOR$$SIG_NO_LUB : NOVALUE,                  ! same as FOR$$SIGNAL_STO except no LUB setup
;    95          0159  1                                                 ! so must pass LUN explicitly.
;    96          0160  1     FOR$$CB_PUSH : JSB_CB_PUSH NOVALUE,         ! push current LUB/ISB/RAB, if any, and allocate LUB/ISB/RAB
;    97          0161  1                                                 ! for this logical unit
;    98          0162  1     FOR$$CB_POP : JSB_CB_POP NOVALUE,           ! Pop I/O system back to previous LUB or indicate
;    99          0163  1                                                 ! no I/O statement is currently being processed.
;   100          0164  1     FOR$$OPEN_KEYWD,                            ! Look up keywords for literal values
;   101          0165  1     FOR$$SIG_FATINT : NOVALUE,                  ! Signal_stop internal error
;   102          0166  1     FOR$$CLOSE_FILE : CALL_CCB;                 ! Close a file
;   103          0167  1
```

```
;  105     0168  1 GLOBAL ROUTINE FOR$OPEN (                        ! FORTRAN OPEN statement
;  106     0169  1         KEYWD,                                   ! keyword code - repeated arguments
;  107     0170  1         INFO                                     ! value of keyword (optional)
;  108     0171  1     ) =                                          ! value is TRUE iff successful, FALSE if error and ERR=
;  109     0172  1
;  110     0173  1 !++
;  111     0174  1 ! ABSTRACT:
;  112     0175  1 !
;  113     0176  1 !       Open file on the specified logical unit (LUN) with
;  114     0177  1 !       attributes specified in the keyword parameters and allocate
;  115     0178  1 !       3 control blocks for use by subsequent I/O statement calls
;  116     0179  1 !       for this LUN.  The 3 control blocks are: Logical Unit
;  117     0180  1 !       Block (LUB), I/O statement block (ISB), and one RMS
;  118     0181  1 !       control block: the RAB.  If a previous CALL ASSIGN
;  119     0182  1 !       or CALL FDBSET has been done all of these control blocks
;  120     0183  1 !       have already been allocated, and a FAB has been
;  121     0184  1 !       allocated to hold the information passed to CALL ASSIGN or
;  122     0185  1 !       CALL FDBSET.
;  123     0186  1 !       An RMS $OPEN or $CONNECT is performed.
;  124     0187  1 !       Then a record buffer is allocated for the LUN.
;  125     0188  1 !
;  126     0189  1 ! FORMAL PARAMETERS:
;  127     0190  1 !
;  128     0191  1 !    The following pair is repeated for each user specified keyword:
;  129     0192  1 !       KEYWD.rlu.v             Contains KEY<7:0>, ARGTYPE<15:8>, and possibly
;  130     0193  1 !                               INFO<31:16>
;  131     0194  1 !       INFO.rlu.v              optional information if need more than
;  132     0195  1 !                               16-bits
;  133     0196  1 !
;  134     0197  1 ! IMPLICIT INPUTS:
;  135     0198  1 !
;  136     0199  1 !       FOR$$A_CUR_LUB          Current active LUB to be pushed
;  137     0200  1 !                               down or 0 if no LUB has an I/O
;  138     0201  1 !                               statement in progress (usual).
;  139     0202  1 !                               Restored on return from FOR$OPEN
;  140     0203  1 !       LUB$V_FAB               1 if FAB allocated by FDBSET, CALL ASSIGN
;  141     0204  1 !       LUB$V_DIRECT            1 if DEFINE FILE done
;  142     0205  1 !       LUB$V_OPENED            1 if unit already opened
;  143     0206  1 !
;  144     0207  1 ! IMPLICIT OUTPUTS:
;  145     0208  1 !
;  146     0209  1 !       LUB$V_READ_ONLY         1 if 'READONLY' present
;  147     0210  1 !       LUB$V_DIRECT            1 if ACCESS = 'DIRECT'
;  148     0211  1 !       LUB$V_APPEND            1 if ACCESS = 'APPEND'
;  149     0212  1 !       LUB$V_OLD_FILE          1 if TYPE = 'OLD'
;  150     0213  1 !       LUB$V_SCRATCH           1 if TYPE = 'SCRATCH'
;  151     0214  1 !       LUB$V_PRINT             1 if DISPOSE = 'PRINT'
;  152     0215  1 !       LUB$V_FIXED             1 if RECORDTYPE = 'FIXED'
;  153     0216  1 !       LUB$V_FORMATTED         1 if FORM = 'FORMATTED' or ommitted
;  154     0217  1 !       LUB$V_UNFORMAT          1 if FORM = 'UNFORMATTED'
;  155     0218  1 !       LUB$A_ASSOC_VAR         adr. of n if ASSOCIATEVARIABLE = n is present
;  156     0219  1 !       LUB$V_ASS_VAR_L         1 if n is longword
;  157     0220  1 !       LUB$W_IFI               RMS internal file id. Needed in case
;  158     0221  1 !                               FORTRAN CLOSE done.
;  159     0222  1 !       LUB$W_RBUF_SIZE         Size in bytes of record buffer allocated.
;  160     0223  1 !
;  161     0224  1 ! COMPLETTION STATUS:
```

```
;  162     0225  1 !
;  163     0226  1 !       TRUE if success, FALSE if failure and ERR= keyword present
;  164     0227  1 !
;  165     0228  1 ! SIDE EFFECTS:
;  166     0229  1 !
;  167     0230  1 !       Allocates LUB/ISB/RAB if not already allocated
;  168     0231  1 !       by CALL ASSIGN, DEFINE FILE, OR CALL FDBSET.
;  169     0232  1 !       SIGNALs or SIGNAL_STOPs the following errors unless ERR=
;  170     0233  1 !       keyword is present: SIGNAL_STOPs FOR$_INCOPECLO (46 =
;  171     0234  1 !       'INCONSISTENT OPEN/CLOSE STATEMENT SPECIFICATIONS')
;  172     0235  1 !       SIGNAL_STOPS FOR$RECIO_OPE (40='RECURSIVE I/O OPERATION')
;  173     0236  1 !       SIGNAL_STOPS FOR$_INVLOGUNI (32='INVALID LOGICAL UNIT NUMBER')
;  174     0237  1 !       See FOR$$OPEN_PROC for other SIGNAL_STOPS.
;  175     0238  1 !
;  176     0239  1 !--
;  177     0240  1
;  178     0241  2     BEGIN
;  179     0242  2
;  180     0243  2     GLOBAL REGISTER
;  181     0244  2         CCB = K_CCB_REG : REF $FOR$CCB_DECL;
;  182     0245  2
;  183     0246  2 !+
;  184     0247  2 ! Use the formal arg list as a VECTOR of blocks; each block = 1 longword.
;  185     0248  2 !-
;  186     0249  2
;  187     0250  2     MAP
;  188     0251  2         KEYWD : BLOCKVECTOR [255, 1];
;  189     0252  2
;  190     0253  2     BUILTIN
;  191     0254  2         ACTUALCOUNT;
;  192     0255  2
;  193     0256  2     LOCAL
;  194     0257  2         NAM_DSC : DSC$DESCRIPTOR,                   ! String descriptor for ASCIZ filename
;  195     0258  2         DEF_DSC : DSC$DESCRIPTOR,                   ! String descriptor for ASCIZ default file name
;  196     0259  2         L_UNWIND_ACTION : VOLATILE,                 ! UNWIND action code for handler
;  197     0260  2         OPEN : VOLATILE VECTOR [OPEN$K_KEY_MAX + 1];    ! open parameter array
;  198     0261  2
;  199     0262  2 !+
;  200     0263  2 ! Establish handler to RESIGNAL or UNWIND if ERR= present
;  201     0264  2 ! depending on OPEN[OPEN$K_ERR].  Pass UNWIND action code.
;  202     0265  2 !-
;  203     0266  2
;  204     0267  2     ENABLE
;  205     0268  2         FOR$$ERR_OPECLO (L_UNWIND_ACTION, OPEN);
;  206     0269  2
;  207     0270  2 !+
;  208     0271  2 ! Set UNWIND cleanup to be a no-operation since LUB/ISB/RAB
;  209     0272  2 ! has not been pushed yet.
;  210     0273  2 !-
;  211     0274  2     L_UNWIND_ACTION = FOR$K_UNWINDNOP;
;  212     0275  2 !+
;  213     0276  2 ! Copy user keyword arglist into array OPEN
;  214     0277  2 ! in cannonical order, so that args may be processed in order
;  215     0278  2 ! If ASCIZ name string, setup NAM_DSC as its descriptor
;  216     0279  2 ! If ASCIZ default name string, setup DEF_DSC as its descriptor
;  217     0280  2 ! SIGNAL_STOP FOR$_INVARGFOR (48='INVALID ARGUMENT TO FORTRAN I/O SYSTEM'),
;  218     0281  2 ! after scanning all parameters and setting up ERR= in OPEN array.
```

```
;  219     0282  2 !-
;  220     0283  2     FOR$$OPECLO_ARG (KEYWD, ACTUALCOUNT (), OPEN, OPEN$K_KEY_MAX, NAM_DSC, DEF_DSC, 1);
;  221     0284  2 !+
;  222     0285  2 ! Allocate LUB/ISB/RAB if not already allocated for this
;  223     0286  2 ! logical unit. Push down if an I/O statement already in progress
;  224     0287  2 ! on another unit. Store new current LUB address in OTS
;  225     0288  2 ! GLOBAL OWN OTS$A_CUR_LUB. SIGNAL_STOP FOR$_RECIO_OPE
;  226     0289  2 ! (40='RECURSIVE I7O OPERATION'). If an I/O statement already
;  227     0290  2 ! in progress for this logical unit. SIGNAL_STOP FOR$_INVLOGUNI
;  228     0291  2 ! (32='INVALID LOGICAL UNIT NUMBER') if logical unit
;  229     0292  2 ! number outside of the allowed range of 0:99 for explicit OPEN.
;  230     0293  2 ! Finally change UNWIND cleanup action to be to pop current LUB/ISB/RAB
;  231     0294  2 ! since it has now been successfully pushed.
;  232     0295  2 ! On return, CCB points to the current control block.
;  233     0296  2 !-
;  234     0297  2     FOR$$CB_PUSH (.OPEN [OPEN$K_UNIT], LUB$K_LUN_MIN);
;  235     0298  2     L_UNWIND_ACTION = FOR$K_UNWINDPOP;
;  236     0299  2 !+
;  237     0300  2 ! If the unit is currently open, call special routine which
;  238     0301  2 ! implements open on a connected unit.
;  239     0302  2 !-
;  240     0303  2
;  241     0304  3     IF (.CCB [LUB$V_OPENED] OR .CCB [LUB$V_DEALLOC])
;  242     0305  2     THEN
;  243     0306  2
;  244     0307  2         IF OPEN_ON_CONNECTED (OPEN, L_UNWIND_ACTION)
;  245     0308  2         THEN
;  246     0309  3             BEGIN
;  247     0310  3 !+
;  248     0311  3 ! No more OPEN processing needed, set IOSTAT and exit.
;  249     0312  3 !-
;  250     0313  3
;  251     0314  4             IF (.OPEN [OPEN$K_IOSTAT] NEQ 0)
;  252     0315  3             THEN
;  253     0316  4                 BEGIN
;  254     0317  4
;  255     0318  5                 IF (.OPEN [OPEN$K_IOSTAT_L])
;  256     0319  4                 THEN
;  257     0320  4                     .OPEN [OPEN$K_IOSTAT] = 0
;  258     0321  4                 ELSE
;  259     0322  5                     BEGIN
;  260     0323  5
;  261     0324  5                     LOCAL
;  262     0325  5                         IOSTAT : REF BLOCK [, BYTE];
;  263     0326  5
;  264     0327  5                     IOSTAT = .OPEN [OPEN$K_IOSTAT];
;  265     0328  5                     IOSTAT [0, 0, 16, 0] = 0;   ! Store one word
;  266     0329  4                     END;
;  267     0330  4
;  268     0331  3                 END;
;  269     0332  3
;  270     0333  3             RETURN 1;                           ! Exit OPEN successfully
;  271     0334  2             END;
;  272     0335  2
;  273     0336  2 !+
;  274     0337  2 ! If DEFINE FILE, CALL FDBSET, or CALL ASSIGN have already been
;  275     0338  2 ! done for this logical unit, SIGNAL_STOP FOR$DUPFILSPE
```

```
;  276     0339  2 ! (21='DUPLICATE FILE SPECIFICATION').
;  277     0340  2 !-
;  278     0341  2
;  279     0342  2     IF ((.CCB [LUB$A_FAB] NEQA 0) OR (.CCB [LUB$V_DIRECT])) THEN FOR$$SIGNAL_STO (FOR$K_DUPFILSPE);
;  280     0343  2
;  281     0344  2 !+
;  282     0345  2 ! Set unwind condition to RET so if an error occurs the file will
;  283     0346  2 ! be closed and the LUB returned (thus freeing up the LUN).
;  284     0347  2 !-
;  285     0348  2     L_UNWIND_ACTION = FOR$K_UNWINDRET;
;  286     0349  2 !+
;  287     0350  2 ! Perform the OPEN - call common procedure with a pointer
;  288     0351  2 ! to the OPEN parameter VECTOR of longword values.
;  289     0352  2 !-
;  290     0353  2     FOR$$OPEN_PROC (OPEN);
;  291     0354  2 !+
;  292     0355  2 ! Pop back previous LUB or indicate that no I/O statement
;  293     0356  2 ! is currently active (OTS$$A_CUR_LUB = 0).
;  294     0357  2 !-
;  295     0358  2     FOR$$CB_POP ();
;  296     0359  2 !+
;  297     0360  2 ! Store success IOSTAT.  If there was an error, the handler would
;  298     0361  2 ! do the store.
;  299     0362  2 !-
;  300     0363  2
;  301     0364  3     IF (.OPEN [OPEN$K_IOSTAT] NEQ 0)
;  302     0365  3     THEN
;  303     0366  3
;  304     0367  3         IF (.OPEN [OPEN$K_IOSTAT_L])
;  305     0368  3         THEN
;  306     0369  3             .OPEN [OPEN$K_IOSTAT] = 0
;  307     0370  3         ELSE
;  308     0371  3             BEGIN
;  309     0372  3
;  310     0373  3             LOCAL
;  311     0374  3                 IOSTAT : REF BLOCK [, BYTE];
;  312     0375  3
;  313     0376  3             IOSTAT = .OPEN [OPEN$K_IOSTAT];
;  314     0377  3             IOSTAT [0, 0, 16, 0] = 0;          ! Store one word
;  315     0378  2             END;
;  316     0379  2
;  317     0380  2 !+
;  318     0381  2 ! Return success
;  319     0382  2 !-
;  320     0383  2     RETURN 1;
;  321     0384  1     END;                                       ! End of FOR$OPEN routine


                                .TITLE  FOR$OPEN FORTRAN OPEN
                                .IDENT  \1-065\

                                .EXTRN  FOR$$ERR_OPECLO
                                .EXTRN  FOR$$OPEN_PROC, FOR$$SIGNAL_STO
                                .EXTRN  FOR$$SIG_NO_LUB
                                .EXTRN  FOR$$CB_PUSH, FOR$$CB_POP
                                .EXTRN  FOR$$OPEN_KEYWD
                                .EXTRN  FOR$$SIG_FATINT
```

```
                                                       .EXTRN  FOR$$CLOSE_FILE

                                                       .PSECT  _FOR$CODE,NOWRT,  SHR,  PIC,2

                              0804 00000               .ENTRY  FOR$OPEN, Save R2,R11              ; 0168
                  5E      88  AE  9E 00002             MOVAB   -120(SP), SP                       ;
                              7E  7C 00006             CLRQ    OPEN                               ; 0241
                          08  AE  7C 00008             CLRQ    OPEN                               ;
                          10  AE  7C 0000B             CLRQ    OPEN                               ;
                          18  AE  7C 0000E             CLRQ    OPEN                               ;
                          20  AE  7C 00011             CLRQ    OPEN                               ;
                          28  AE  7C 00014             CLRQ    OPEN                               ;
                          30  AE  7C 00017             CLRQ    OPEN                               ;
                          38  AE  7C 0001A             CLRQ    OPEN                               ;
                          40  AE  7C 0001D             CLRQ    OPEN                               ;
                          48  AE  7C 00020             CLRQ    OPEN                               ;
                          50  AE  7C 00023             CLRQ    OPEN                               ;
                          58  AE  7C 00026             CLRQ    OPEN                               ;
                          60  AE  7C 00029             CLRQ    OPEN                               ;
                          68  AE  7C 0002C             CLRQ    OPEN                               ;
                  6D    0081  CF  DE 0002F             MOVAL   8$, (FP)                           ;
          6C      AE          01  D0 00034             MOVL    #1, L_UNWIND_ACTION                ; 0274
                              01  DD 00038             PUSHL   #1                                 ; 0283
                          74  AE  9F 0003A             PUSHAB  DEF_DSC                            ;
                          F8  AD  9F 0003D             PUSHAB  NAM_DSC                            ;
                              1A  DD 00040             PUSHL   #26                                ;
                          10  AE  9F 00042             PUSHAB  OPEN                               ;
                  7E          6C  9A 00045             MOVZBL  (AP), -(SP)                        ;
                          04  AC  9F 00048             PUSHAB  KEYWD                              ;
       0000V      CF          07  FB 0004B             CALLS   #7, FOR$$OPECLO_ARG                ;
                              50  D4 00050             CLRL    R0                                 ; 0297
                  52      04  AE  D0 00052             MOVL    OPEN+4, R2                         ;
                   00000000G  00  16 00056             JSB     FOR$$CB_PUSH                       ;
                          6C  AE  D4 0005C             CLRL    L_UNWIND_ACTION                    ; 0298
                  05      FC  AB  E8 0005F             BLBS    -4(CCB), 1$                        ; 0304
  0E      FF      AB          04  E1 00063             BBC     #4, -1(CCB), 2$                    ;
                          6C  AE  9F 00068 1$:         PUSHAB  L_UNWIND_ACTION                    ; 0307
                          04  AE  9F 0006B             PUSHAB  OPEN                               ;
       0000V      CF          02  FB 0006E             CALLS   #2, OPEN_ON_CONNECTED              ;
                  26          50  E8 00073             BLBS    R0, 5$                             ;
                          E8  AB  D5 00076 2$:         TSTL    -24(CCB)                           ; 0342
                              05  12 00079             BNEQ    3$                                 ;
  09      FC      AB          04  E1 0007B             BBC     #4, -4(CCB), 4$                    ;
                              15  DD 00080 3$:         PUSHL   #21                                ;
   00000000G      00          01  FB 00082             CALLS   #1, FOR$$SIGNAL_STO                ;
          6C      AE          02  D0 00089 4$:         MOVL    #2, L_UNWIND_ACTION                ; 0348
                              5E  DD 0008D             PUSHL   SP                                 ; 0353
   00000000G      00          01  FB 0008F             CALLS   #1, FOR$$OPEN_PROC                 ;
                   00000000G  00  16 00096             JSB     FOR$$CB_POP                        ; 0358
                          58  AE  D5 0009C 5$:         TSTL    OPEN+88                            ; 0364
                              0F  13 0009F             BEQL    7$                                 ;
                  05      64  AE  E9 000A1             BLBC    OPEN+100, 6$                       ; 0367
                          58  BE  D4 000A5             CLRL    @OPEN+88                           ; 0369
                              06  11 000A8             BRB     7$                                 ;
                  50      58  AE  D0 000AA 6$:         MOVL    OPEN+88, IOSTAT                    ; 0376
                              60  B4 000AE             CLRW    (IOSTAT)                           ; 0377
                  50          01  D0 000B0 7$:         MOVL    #1, R0                             ; 0383
```

```
                                    04 000B3          RET                                                 ; 0384
                                0000 000B4 8$:        .WORD   Save nothing                                ; 0241
                     50     08  AC  D0 000B6          MOVL    8(AP), R0                                   ;
                     50     04  A0  D0 000BA          MOVL    4(R0), R0                                   ;
                            80  A0  9F 000BE          PUSHAB  OPEN                                        ;
                            EC  A0  9F 000C1          PUSHAB  L_UNWIND_ACTION                             ;
                                02  DD 000C4          PUSHL   #2                                          ;
                                5E  DD 000C6          PUSHL   SP                                          ;
                         7E     04  AC  7D 000C8          MOVQ    4(AP), -(SP)                            ;
               00000000G 00             03  FB 000CC          CALLS   #3, FOR$$ERR_OPECLO                 ;
                                        04 000D3          RET                                             ;

; Routine Size:  212 bytes,    Routine Base:  _FOR$CODE + 0000

;  322          0385  1
```

```
;  324          0386  1 GLOBAL ROUTINE FOR$$OPECLO_ARG (                      ! FORTRAN copy OPEN/CLOSE args
;  325          0387  1         KEYWD_ADR,                                   ! Adr. of first keyword arg
;  326          0388  1         ACTUAL_COUNT,                                ! No. of actual parameters in arg list
;  327          0389  1         OPEN_ADR,                                    ! Adr. of array to store keyword values
;  328          0390  1         KEY_MAX,                                     ! Max. value of keyword parameter
;  329          0391  1         NAM_DSC_ADR,                                 ! Adr. of descriptor for name string
;  330          0392  1         DEF_DSC_ADR,                                 ! Adr. of descriptor for default name string
;  331          0393  1         OPEN_FLAG,                                   ! True if OPEN (not CLOSE)
;  332          0394  1         VAR_LENGTHS                                  ! lengths in bits of keyword variables
;  333          0395  1     ) : NOVALUE =
;  334          0396  1
;  335          0397  1 !++
;  336          0398  1 ! ABSTRACT:
;  337          0399  1 !
;  338          0400  1 !       Routine to copy keyword OPEN/CLOSE parameters
;  339          0401  1 !       into an array for sequential processing in cannonical order.
;  340          0402  1 !       Note: LUB cannot be located until all OPEN arguments are scanned and UNIT=n found.
;  341          0403  1 !
;  342          0404  1 ! FORMAL PARAMETERS:
;  343          0405  1 !
;  344          0406  1 !       KEYWD_ADR.rlu.ra        Address of first keyword
;  345          0407  1 !                               in user arg list
;  346          0408  1 !       ACTUAL_COUNT.rlu.v      Count of no. of users args
;  347          0409  1 !       OPEN_ADR.wlu.ra         Adr. of array to write keyword values
;  348          0410  1 !       KEY_MAX.rlu.v   Max. OPEN/CLOSE keyword value
;  349          0411  1 !       NAM_DSC_ADR     Adr. of a descriptor if ASCIZ name string given by user
;  350          0412  1 !       DEF_DSC_ADR     Adr. of a descriptor if ASCIZ default name string given by user
;  351          0413  1 !                       Descriptors must be allocated by caller
;  352          0414  1 !                       not called procedure.
;  353          0415  1 !       OPEN_FLAG       = 1 if this call is from OPEN, 0 from CLOSE.
;  354          0416  1 !                       Only allocate a LUN if from OPEN.
;  355          0417  1 !       VAR_LENGTHS     A byte vector into which are inserted the lengths
;  356          0418  1 !                       in bits of the keyword variables.  This is used
;  357          0419  1 !                       by FOR$INQUIRE only.
;  358          0420  1 !
;  359          0421  1 ! IMPLICIT INPUTS:
;  360          0422  1 !
;  361          0423  1 !       NONE
;  362          0424  1 !
;  363          0425  1 ! IMPLICIT OUTPUTS:
;  364          0426  1 !
;  365          0427  1 !       NONE
;  366          0428  1 !
;  367          0429  1 ! COMPLETITION STATUS:
;  368          0430  1 !
;  369          0431  1 !       NONE
;  370          0432  1 !
;  371          0433  1 ! SIDE EFFECTS:
;  372          0434  1 !
;  373          0435  1 !       SIGNAL_STOPs FOR$_INVARGFOR (48='INVALID ARGUMENT TO FORTRAN I/O SYSTEM')
;  374          0436  1 !       if keyword parameter is out of range, but only after all parameters
;  375          0437  1 !       are scanned so that ERR= parameter, if present, has been setup in array OPEN_ADR.
;  376          0438  1 !       Uses FOR$$SIG_NO_LUB to signal, since no LUB setup yet
;  377          0439  1 !       so logical unit number must be passed explicitly on errors.
;  378          0440  1 !--
;  379          0441  1
;  380          0442  2     BEGIN
```

```
;  381     0443  2
;  382     0444  2      MAP
;  383     0445  2          KEYWD_ADR : REF BLOCKVECTOR [100, 1],    ! Vector of blocks, each block
;  384     0446  2                                                   ! is one longword.
;  385     0447  2          OPEN_ADR : REF VECTOR [OPEN$K_KEY_MAX + 1, LONG],       ! Vector to receive cannonical ordering
;  386     0448  2                                                   ! of users parameter values.
;  387     0449  2          NAM_DSC_ADR : REF DSC$DESCRIPTOR,        ! string descriptor to use in case ASCIZ file name
;  388     0450  2          DEF_DSC_ADR : REF DSC$DESCRIPTOR,        ! string descriptor to use in case ASCIZ default file name
;  389     0451  2          VAR_LENGTHS : REF VECTOR [INQ$K_KEY_MAX + 1, BYTE];     ! Variable lengths
;  390     0452  2
;  391     0453  2      LOCAL
;  392     0454  2          V_ARG_KEY_ERR,                           ! error flag, 1 if ARG or KEY out of range
;  393     0455  2          V_KEY_VAL_ERR,                           ! error flag, 1 if keyword incorrect
;  394     0456  2          UNIT_ADDR,                               ! Address of UNIT variable
;  395     0457  2          UNIT_TYPE;                               ! Type of varaible: w or L
;  396     0458  2
;  397     0459  2  !+
;  398     0460  2  ! Clear OPEN or CLOSE parameter array and clear flag
;  399     0461  2  !-
;  400     0462  2      FILL_VAL (0, .KEY_MAX + 1, .OPEN_ADR);
;  401     0463  2      V_ARG_KEY_ERR = 0;
;  402     0464  2      V_KEY_VAL_ERR = 0;
;  403     0465  2      UNIT_TYPE = 0;
;  404     0466  2      UNIT_ADDR = 0;
;  405     0467  2  !+
;  406     0468  2  ! Scan actual keyword parameter list (KEYWD_ADR) and copy (sign extend)
;  407     0469  2  ! associated information to formal array OPEN_ADR of longwords ordered
;  408     0470  2  ! by parameter dependencies, i. e., sort by KEY.
;  409     0471  2  !-
;  410     0472  2
;  411     0473  2      INCR I FROM 0 TO .ACTUAL_COUNT - 1 DO
;  412     0474  2  !+
;  413     0475  2  ! Set longword value to sign extension of each type of OPEN/CLOSE
;  414     0476  2  ! parameter present to: Bits 31:16 of this actual, next
;  415     0477  2  ! actual, or location specified by next actual depending
;  416     0478  2  ! on the type of OPEN argument (OPEN$B_ARG_TYPE).
;  417     0479  2  ! If ARGTYPE or KEY code is not one of defined values, set error flag and keep scanning
;  418     0480  2  ! to see if ERR= is present so error handler will handle properly.
;  419     0481  2  ! error FOR$_INVARGFOR (48='INVALID ARGUMENT TO FORTRAN I/O SYSTEM')
;  420     0482  2  !-
;  421     0483  3          BEGIN
;  422     0484  3
;  423     0485  3          LOCAL
;  424     0486  3              K,                                   ! temp value of KEY
;  425     0487  3              V;                                   ! temp value of value to be stored
;  426     0488  3
;  427     0489  3          K = .KEYWD_ADR [.I, OPEN$B_KEY];
;  428     0490  3          V =
;  429     0491  4          BEGIN
;  430     0492  4
;  431     0493  4          CASE .KEYWD_ADR [.I, OPEN$B_ARG_TYPE] FROM 0 TO OPEN$K_ARG_MAX OF
;  432     0494  4              SET
;  433     0495  4
;  434     0496  4              [OPEN$K_ARG_NULL] :
;  435     0497  4  !+
;  436     0498  4  ! keyword with no value - make value be 1
;  437     0499  4  ! to distinguish from not present.
```

```
;  438     0500  4 !-
;  439     0501  4                     1;
;  440     0502  4
;  441     0503  4                 [OPEN$K_ARG_LIT, OPEN$K_ARG_W_V] :
;  442     0504  4 !+
;  443     0505  4 ! literal or word-by-value - bits <31:16> is value
;  444     0506  4 ! sign extend to full machine value
;  445     0507  4 !-
;  446     0508  4                     .KEYWD_ADR [.I, OPEN$W_INFO];
;  447     0509  4
;  448     0510  4                 [OPEN$K_ARG_W_R] :
;  449     0511  4 !+
;  450     0512  4 ! Word by reference - use adr. in next longword
;  451     0513  4 ! sign extend word to longword
;  452     0514  4 !-
;  453     0515  5                     BEGIN
;  454     0516  5
;  455     0517  6                     IF (.K EQLU OPEN$K_UNIT)
;  456     0518  5                     THEN
;  457     0519  5 !+
;  458     0520  5 ! Remember UNIT's address and type in case we must provide it
;  459     0521  5 !-
;  460     0522  5
;  461     0523  6                         IF (.UNIT_TYPE NEQ 0)
;  462     0524  5                         THEN
;  463     0525  5                             V_ARG_KEY_ERR = 1
;  464     0526  5                         ELSE
;  465     0527  6                             BEGIN
;  466     0528  6 !+
;  467     0529  6 ! This is the first time through here
;  468     0530  6 !-
;  469     0531  6                             UNIT_TYPE = DSC$K_DTYPE_W;
;  470     0532  6                             UNIT_ADDR = .KEYWD_ADR [.I + 1, OPEN$A_VALUE];
;  471     0533  5                             END;
;  472     0534  5
;  473     0535  6                     IF ((.K EQLU OPEN$K_ASSOCIAT) OR (.K EQLU OPEN$K_IOSTAT))
;  474     0536  5                     THEN
;  475     0537  5 !+
;  476     0538  5 ! For the associated variable or IOSTAT we want the address of the value, not the
;  477     0539  5 ! value itself.
;  478     0540  5 !-
;  479     0541  5                         .KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE]     !
;  480     0542  5                     ELSE
;  481     0543  5
;  482     0544  6                         IF (.K GTR OPEN$K_KEY_MAX)
;  483     0545  5                         THEN
;  484     0546  6                             BEGIN
;  485     0547  6                             VAR_LENGTHS [.K] = 16;  ! Signify word
;  486     0548  6                             .KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE]
;  487     0549  6                             END
;  488     0550  5                         ELSE
;  489     0551  5                             .(.KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE])<0, %BPVAL/2, 1>
;  490     0552  5
;  491     0553  4                     END;
;  492     0554  4
;  493     0555  4                 [OPEN$K_ARG_L_R] :
;  494     0556  4 !+
```

```
;  495     0557  4 ! Longword by-reference-next parameter slot contains adr. of value
;  496     0558  4 !-
;  497     0559  5                     BEGIN
;  498     0560  5
;  499     0561  6                     IF (.K EQLU OPEN$K_UNIT)
;  500     0562  5                     THEN
;  501     0563  5 !+
;  502     0564  5 ! Remember the address and type of the variable which holds the UNIT
;  503     0565  5 ! in case we must compute the LUN value.
;  504     0566  5 !-
;  505     0567  5
;  506     0568  6                         IF (.UNIT_TYPE NEQ 0)
;  507     0569  5                         THEN
;  508     0570  5                             V_ARG_KEY_ERR = 1
;  509     0571  5                         ELSE
;  510     0572  6                             BEGIN
;  511     0573  6 !+
;  512     0574  6 ! This is the first time through here.
;  513     0575  6 !-
;  514     0576  6                             UNIT_TYPE = DSC$K_DTYPE_L;
;  515     0577  6                             UNIT_ADDR = .KEYWD_ADR [.I + 1, OPEN$A_VALUE];
;  516     0578  5                             END;
;  517     0579  5
;  518     0580  6                     IF ((.K EQLU OPEN$K_ASSOCIAT) OR (.K EQLU OPEN$K_IOSTAT))
;  519     0581  5                     THEN
;  520     0582  5 !+
;  521     0583  5 ! For the associated variable or IOSTAT we want the address of the variable, not
;  522     0584  5 ! its value.  Also, we must mark that it occupies a longword.
;  523     0585  5 !-
;  524     0586  6                         BEGIN
;  525     0587  6
;  526     0588  7                         IF (.K EQLU OPEN$K_ASSOCIAT)
;  527     0589  6                         THEN
;  528     0590  6                             OPEN_ADR [OPEN$K_ASSOC_L] = 1
;  529     0591  6                         ELSE
;  530     0592  6                             OPEN_ADR [OPEN$K_IOSTAT_L] = 1;
;  531     0593  6
;  532     0594  6                         .KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE]
;  533     0595  6                         END
;  534     0596  5                     ELSE
;  535     0597  5
;  536     0598  6                         IF (.K GTR OPEN$K_KEY_MAX)
;  537     0599  5                         THEN
;  538     0600  6                             BEGIN
;  539     0601  6                             VAR_LENGTHS [.K] = 32;  ! Signify longword
;  540     0602  6                             .KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE] ! Address for INQUIRE
;  541     0603  6                             END
;  542     0604  5                         ELSE
;  543     0605  5                             ..KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE]
;  544     0606  5
;  545     0607  4                     END;
;  546     0608  4
;  547     0609  4                 [OPEN$K_ARG_L_V, OPEN$K_ARG_ZI] :
;  548     0610  4 !+
;  549     0611  4 ! Longword by value or procedure adr.
;  550     0612  4 !-
;  551     0613  4                     .KEYWD_ADR [(I = .I + 1), OPEN$G_VALUE];
```

```
;  552     0614  4
;  553     0615  4                [OPEN$K_ARG_T_DS] :
;  554     0616  4 !+
;  555     0617  4 ! Address of string descriptor.
;  556     0618  4 !-
;  557     0619  4
;  558     0620  4                    IF .K EQLU OPEN$K_NAME OR .K EQLU OPEN$K_DEFAULTF
;  559     0621  4                    THEN
;  560     0622  4                        .KEYWD_ADR [(I = .I + 1), OPEN$G_VALUE]
;  561     0623  4                    ELSE
;  562     0624  5                        BEGIN
;  563     0625  5
;  564     0626  5                        LOCAL
;  565     0627  5                            V;                          ! Returned value
;  566     0628  5
;  567     0629  5                        V = FOR$$OPEN_KEYWD (.K, .KEYWD_ADR [.I + 1, OPEN$G_VALUE]);
;  568     0630  5                        I = .I + 1;
;  569     0631  5
;  570     0632  5                        CASE .V FROM -1 TO 0 OF
;  571     0633  5                            SET
;  572     0634  5
;  573     0635  5                            [-1] :                      ! Invalid keyword for this type
;  574     0636  6                                BEGIN
;  575     0637  6                                V_ARG_KEY_ERR = 1;
;  576     0638  6                                0
;  577     0639  5                                END;
;  578     0640  5
;  579     0641  5                            [0] :                       ! Keyword value error
;  580     0642  6                                BEGIN
;  581     0643  6                                V_KEY_VAL_ERR = 1;
;  582     0644  6                                0
;  583     0645  5                                END;
;  584     0646  5
;  585     0647  5                            [OUTRANGE] :                ! Ok
;  586     0648  5                                .V;
;  587     0649  5                            TES
;  588     0650  5
;  589     0651  4                        END;
;  590     0652  4
;  591     0653  4                [OPEN$K_ARG_TZ_R] :
;  592     0654  4 !+
;  593     0655  4 ! Address of array of ASCIZ characters.
;  594     0656  4 ! Next parameter slot contains address of first byte of string
;  595     0657  4 ! If this is FILE or DEFAULTFILE, store length and address of string in
;  596     0658  4 ! its respective descriptor.
;  597     0659  4 ! Else SIGNAL_STOP FOR$_INVARGFOR (48='INVALID ARGUMENT TO FORTRAN I/O SYSTEM')
;  598     0660  4 !-
;  599     0661  4
;  600     0662  5                    IF (.K EQLU OPEN$K_NAME)
;  601     0663  4                    THEN
;  602     0664  5                        BEGIN
;  603     0665  5
;  604     0666  5                        LOCAL
;  605     0667  5                            P;                          ! char. pointer to null char or 0
;  606     0668  5
;  607     0669  5                        NAM_DSC_ADR [DSC$A_POINTER] = .KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE];
;  608     0670  5                        P = CH$FIND_CH (OPEN$K_STR_MAX, .NAM_DSC_ADR [DSC$A_POINTER], 0);
```

```
;  609     0671  6                    NAM_DSC_ADR [DSC$W_LENGTH] = (IF .P NEQ 0 THEN CH$DIFF (.P, .NAM_DSC_ADR [DSC$A_POINTER]
;  610     0672  5                    ELSE OPEN$K_STR_MAX);
;  611     0673  5                    .NAM_DSC_ADR                    ! value of the CASE-expr is adr. of descr.
;  612     0674  5                    END
;  613     0675  5                ELSE IF (.K EQLU OPEN$K_DEFAULTF)
;  614     0676  4                THEN
;  615     0677  5                    BEGIN
;  616     0678  5
;  617     0679  5                    LOCAL
;  618     0680  5                        P;                          ! char. pointer to null char or 0
;  619     0681  5
;  620     0682  5                    DEF_DSC_ADR [DSC$A_POINTER] = .KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE];
;  621     0683  5                    P = CH$FIND_CH (OPEN$K_STR_MAX, .DEF_DSC_ADR [DSC$A_POINTER], 0);
;  622     0684  6                    DEF_DSC_ADR [DSC$W_LENGTH] = (IF .P NEQ 0 THEN CH$DIFF (.P, .DEF_DSC_ADR [DSC$A_POINTER]
;  623     0685  5                    ELSE OPEN$K_STR_MAX);
;  624     0686  5                    .DEF_DSC_ADR                    ! value of the CASE-expr is adr. of descr.
;  625     0687  5                    END
;  626     0688  4                ELSE
;  627     0689  4            !+
;  628     0690  4            ! ASCIZ string not file name or default file name, just skip next
;  629     0691  4            ! longword and flag error
;  630     0692  4            !-
;  631     0693  5                    BEGIN
;  632     0694  5                    I = .I + 1;
;  633     0695  5                    V_ARG_KEY_ERR = 1;
;  634     0696  5                    0                               ! value of the CASE-expr is 0 if error
;  635     0697  4                    END;
;  636     0698  4
;  637     0699  4                [OPEN$K_ARG_INLN] :
;  638     0700  4            !+
;  639     0701  4            ! Sublist in-line with argument list
;  640     0702  4            !-
;  641     0703  5                    BEGIN
;  642     0704  5
;  643     0705  5                    LOCAL
;  644     0706  5                        ADDR,
;  645     0707  5                        COUNT;
;  646     0708  5
;  647     0709  5                    COUNT = .KEYWD_ADR [.I, OPEN$W_INFO];
;  648     0710  5                    ADDR = KEYWD_ADR [.I, OPEN$B_KEY];
;  649     0711  5                    I = .I + .COUNT;
;  650     0712  5                    .ADDR
;  651     0713  4                    END;
;  652     0714  4
;  653     0715  4                [OPEN$K_ARG_B_R] :
;  654     0716  4            !+
;  655     0717  4            ! Byte variable by reference
;  656     0718  4            ! Used only by FOR$INQUIRE
;  657     0719  4            !-
;  658     0720  5                    BEGIN
;  659     0721  5
;  660     0722  6                    IF (.K GTR OPEN$K_KEY_MAX)
;  661     0723  5                    THEN
;  662     0724  6                        BEGIN
;  663     0725  6                        VAR_LENGTHS [.K] = 8;       ! Signify byte
;  664     0726  6                        .KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE]
;  665     0727  6                        END
```

```
;  666     0728  5                  ELSE
;  667     0729  5                      ..KEYWD_ADR [(I = .I + 1), OPEN$A_VALUE]
;  668     0730  5
;  669     0731  4                  END;
;  670     0732  4
;  671     0733  4              [INRANGE, OUTRANGE] :
;  672     0734  4  !+
;  673     0735  4  ! If KEY is out of range, set error flag (V_ARG_KEY_ERR) and
;  674     0736  4  ! keep scanning to see if ERR= is present or not.
;  675     0737  4  !-
;  676     0738  5                  BEGIN
;  677     0739  5                  V_ARG_KEY_ERR = 1;
;  678     0740  5                  0                                   ! Store 0
;  679     0741  4                  END;
;  680     0742  4              TES
;  681     0743  4
;  682     0744  3          END;                                        ! End of CASE on ARG_TYPE.
;  683     0745  3  !+
;  684     0746  3  ! If KEY value is in range, store in cannonical array OPEN_ADR,
;  685     0747  3  ! else set error flag and keep scanning to see if ERR= is present
;  686     0748  3  ! so error handler will handle properly when signaled.
;  687     0749  3  ! Note: I advanced correctly (by 1 or 2) depending on ARGTYPE
;  688     0750  3  ! even though KEY is not one of the defined ones.
;  689     0751  3  !-
;  690     0752  3
;  691     0753  3          IF ((.K LEQU .KEY_MAX) OR (.K EQLU OPEN$K_IOSTAT)) THEN OPEN_ADR [.K] = .V ELSE V_ARG_KEY_ERR = 1;
;  692     0754  3
;  693     0755  2          END;                                        ! End of loop
;  694     0756  2
;  695     0757  2  !+
;  696     0758  2  ! Check for any errors during scan.
;  697     0759  2  ! If yes, SIGNAL_STOP FOR$_INVARGFOR (48='INVALID ARGUMENT TO FORTRAN I/O SYSTEM')
;  698     0760  2  !-
;  699     0761  2
;  700     0762  2      IF .V_ARG_KEY_ERR THEN FOR$$SIG_NO_LUB (FOR$K_INVARGFOR, .OPEN_ADR [OPEN$K_UNIT]);
;  701     0763  2
;  702     0764  2      IF .V_KEY_VAL_ERR THEN FOR$$SIG_NO_LUB (FOR$K_KEYVALERR, .OPEN_ADR [OPEN$K_UNIT]);
;  703     0765  2
;  704     0766  2
;  705     0767  2      RETURN;                                         ! Return from FOR$$OPECLO_ARG routine
;  706     0768  1      END;                                            ! End of FOR$$OPECLO_ARG routine


                                    0FFC 00000          .ENTRY  FOR$$OPECLO_ARG, Save R2,R3,R4,R5,R6,R7,R8,-; 0386
                                                                R9,R10,R11
                                5E  04  C2 00002        SUBL2   #4, SP
                                5B  10  AC D0 00005     MOVL    KEY_MAX, R11                                  ; 0462
                          50    5B      02 78 00009     ASHL    #2, R11, R0
                                50      04 C0 0000D     ADDL2   #4, R0
                                57  0C  AC D0 00010     MOVL    OPEN_ADR, R7
                    50    00    6E      00 2C 00014     MOVC5   #0, (SP), #0, R0, (R7)
                                          67  00019
                                          6E D4 0001A   CLRL    V_KEY_VAL_ERR                                 ; 0464
                                          58 7C 0001C   CLRQ    UNIT_TYPE                                     ; 0465
```

```
                                    5A  D4 0001E          CLRL    UNIT_ADDR                                       ; 0466
                              55    04  AC D0 00020          MOVL    KEYWD_ADR, R5                                   ; 0489
                              52        01 CE 00024          MNEGL   #1, I
                                  0175  31 00027          BRW     42$
                              50      6542 DE 0002A 15$:     MOVAL   (R5)[I], R0
                              53        60 9A 0002E          MOVZBL  (R0), K
                        0A    00    01  A0 8F 00031          CASEB   1(R0), #0, #10                                  ; 0493
0025                    001E  001E      0019 00036 25$:     .WORD   3$-2$,-
00AE                    00DA  005C      0145 0003E          4$-2$,-
                        013B  012F      0145 00046          4$-2$,-
                                                            6$-2$,-
                                                            37$-2$,-
                                                            10$-2$,-
                                                            25$-2$,-
                                                            20$-2$,-
                                                            37$-2$,-
                                                            34$-2$,-
                                                            36$-2$
                                  010F  31 0004C          BRW     32$                                             ; 0739
                              54        01 D0 0004F 35$:     MOVL    #1, V                                           ; 0493
                                        04 11 00052          BRB     5$
                              54    02  A0 32 00054 45$:     CVTWL   2(R0), V                                        ; 0508
                                  0131  31 00058 55$:     BRW     39$
                              01        53 D1 0005B 65$:     CMPL    K, #1                                           ; 0517
                                        11 12 0005E          BNEQ    8$
                                        58 D5 00060          TSTL    UNIT_TYPE                                       ; 0523
                                        05 13 00062          BEQL    7$
                              59        01 D0 00064          MOVL    #1, V_ARG_KEY_ERR                               ; 0525
                                        08 11 00067          BRB     8$
                              58        07 D0 00069 75$:     MOVL    #7, UNIT_TYPE                                   ; 0531
                              5A  04  A542 D0 0006C          MOVL    4(R5)[I], UNIT_ADDR                             ; 0532
                              11        53 D1 00071 85$:     CMPL    K, #17                                          ; 0535
                                        76 13 00074          BEQL    21$
                              16        53 D1 00076          CMPL    K, #22
                                        71 13 00079          BEQL    21$
                              1A        53 D1 0007B          CMPL    K, #26                                          ; 0544
                                        07 15 0007E          BLEQ    9$
                        20 BC43         10 90 00080          MOVB    #16, @VAR_LENGTHS[K]                            ; 0547
                                        4A 11 00085          BRB     18$                                             ; 0548
                                        52 D6 00087 95$:     INCL    I                                               ; 0551
                              50      6542 D0 00089          MOVL    (R5)[I], R0
                              50        60 32 0008D          CVTWL   (R0), R0
                                        77 11 00090          BRB     23$                                             ; 0544
                              01        53 D1 00092 105$:    CMPL    K, #1                                           ; 0561
                                        11 12 00095          BNEQ    12$
                                        58 D5 00097          TSTL    UNIT_TYPE                                       ; 0568
                                        05 13 00099          BEQL    11$
                              59        01 D0 0009B          MOVL    #1, V_ARG_KEY_ERR                               ; 0570
                                        08 11 0009E          BRB     12$
                              58        08 D0 000A0 115$:    MOVL    #8, UNIT_TYPE                                   ; 0576
                              5A  04  A542 D0 000A3          MOVL    4(R5)[I], UNIT_ADDR                             ; 0577
                                        50 D4 000A8 125$:    CLRL    R0                                              ; 0580
                              11        53 D1 000AA          CMPL    K, #17
                                        04 12 000AD          BNEQ    13$
                                        50 D6 000AF          INCL    R0
                                        05 11 000B1          BRB     14$
                              16        53 D1 000B3 135$:    CMPL    K, #22
```

```
                                 0F 12 000B6      BNEQ    17$                          ;
                          05     50 E9 000B8 14$: BLBC    R0, 15$                      ; 0588
                          67     01 D0 000BB      MOVL    #1, (R7)                     ; 0590
                                 04 11 000BE      BRB     16$                          ;
                    64    A7     01 D0 000C0 15$: MOVL    #1, 100(R7)                  ; 0592
                               00B4 31 000C4 16$: BRW     37$                          ; 0594
                          1A     53 D1 000C7 17$: CMPL    K, #26                       ; 0598
                                 0D 15 000CA      BLEQ    19$                          ;
                    20 BC43      20 90 000CC      MOVB    #32, @VAR_LENGTHS[K]         ; 0601
                                 52 D6 000D1 18$: INCL    I                            ; 0602
                          50   6542 D0 000D3      MOVL    (R5)[I], R0                  ;
                                 30 11 000D7      BRB     23$                          ;
                                 52 D6 000D9 19$: INCL    I                            ; 0605
                          50   6542 D0 000DB      MOVL    (R5)[I], R0                  ;
                          50     60 D0 000DF      MOVL    (R0), R0                     ;
                                 25 11 000E2      BRB     23$                          ; 0598
                          0E     53 D1 000E4 20$: CMPL    K, #14                       ; 0620
                                 DB 13 000E7      BEQL    16$                          ;
                          1A     53 D1 000E9      CMPL    K, #26                       ;
                                 D6 13 000EC 21$: BEQL    16$                          ;
                            04 A542 DD 000EE      PUSHL   4(R5)[I]                     ; 0629
                                 53 DD 000F2      PUSHL   K                            ;
              00000000G   00     02 FB 000F4      CALLS   #2, FOR$$OPEN_KEYWD          ;
                                 52 D6 000F9      INCL    I                            ; 0630
           01 FFFFFFFF    8F     50 CF 000FD      CASEL   V, #-1, #1                   ; 0632
                        0006   0059    00105 22$: .WORD   32$-22$,-                    ;
                                                          24$-22$                      ;
                                 61 11 00109 23$: BRB     35$                          ; 0648
                          6E     01 D0 0010B 24$: MOVL    #1, V_KEY_VAL_ERR            ; 0643
                                 51 11 0010E      BRB     33$                          ; 0642
                          0E     53 D1 00110 25$: CMPL    K, #14                       ; 0662
                                 16 12 00113      BNEQ    26$                          ;
                          56   14 AC D0 00115     MOVL    NAM_DSC_ADR, R6              ; 0669
                                 52 D6 00119      INCL    I                            ;
                    04    A6   6542 D0 0011B      MOVL    (R5)[I], 4(R6)               ;
   04   B6      0064      8F     00 3A 00120      LOCC    #0, #100, @4(R6)             ; 0670
                                 1B 13 00127      BEQL    27$                          ;
                                 1B 11 00129      BRB     28$                          ; 0671
                          1A     53 D1 0012B 26$: CMPL    K, #26                       ; 0675
                                 2C 12 0012E      BNEQ    31$                          ;
                          56   18 AC D0 00130     MOVL    DEF_DSC_ADR, R6              ; 0682
                                 52 D6 00134      INCL    I                            ;
                    04    A6   6542 D0 00136      MOVL    (R5)[I], 4(R6)               ;
   04   B6      0064      8F     00 3A 0013B      LOCC    #0, #100, @4(R6)             ; 0683
                                 02 12 00142      BNEQ    28$                          ;
                                 51 D4 00144 27$: CLRL    R1                           ;
                                 51 D5 00146 28$: TSTL    P                            ; 0684
                                 06 13 00148      BEQL    29$                          ;
                          51   04 A6 C2 0014A     SUBL2   4(R6), R1                    ;
                                 04 11 0014E      BRB     30$                          ;
                          51   64 8F 9A 00150 29$: MOVZBL #100, R1                     ;
                          66     51 B0 00154 30$: MOVW    R1, (R6)                     ;
                          54     56 D0 00157      MOVL    R6, V                        ; 0686
                                 30 11 0015A      BRB     39$                          ;
                                 52 D6 0015C 31$: INCL    I                            ; 0694
                          59     01 D0 0015E 32$: MOVL    #1, V_ARG_KEY_ERR            ; 0695
                                 54 D4 00161 33$: CLRL    V                            ; 0693
```

```
                    27  11 00163        BRB     39$                             ; 0662
        51      02  A0  32 00165 34$:   CVTWL   2(R0), COUNT                    ; 0709
        52          51  C0 00169        ADDL2   COUNT, I                        ; 0711
        54          50  D0 0016C 35$:   MOVL    ADDR, V                         ; 0712
                    1B  11 0016F        BRB     39$                             ;
        1A          53  D1 00171 36$:   CMPL    K, #26                          ; 0722
                    0D  15 00174        BLEQ    38$                             ;
   20 BC43          08  90 00176        MOVB    #8, @VAR_LENGTHS[K]             ; 0725
                    52  D6 0017B 37$:   INCL    I                               ; 0726
        54        6542  D0 0017D        MOVL    (R5)[I], V                      ;
                    09  11 00181        BRB     39$                             ;
                    52  D6 00183 38$:   INCL    I                               ; 0729
        50        6542  D0 00185        MOVL    (R5)[I], R0                     ;
        54          60  D0 00189        MOVL    (R0), V                         ;
        5B          53  D1 0018C 39$:   CMPL    K, R11                          ; 0753
                    05  1B 0018F        BLEQU   40$                             ;
        16          53  D1 00191        CMPL    K, #22                          ;
                    06  12 00194        BNEQ    41$                             ;
      6743          54  D0 00196 40$:   MOVL    V, (R7)[K]                      ;
                    03  11 0019A        BRB     42$                             ;
        59          01  D0 0019C 41$:   MOVL    #1, V_ARG_KEY_ERR               ;
  02    52      08  AC  F2 0019F 42$:   AOBLSS  ACTUAL_COUNT, I, 43$            ; 0473
                    03  11 001A4        BRB     44$                             ;
                  FE81  31 001A6 43$:   BRW     1$                              ;
        0C          59  E9 001A9 44$:   BLBC    V_ARG_KEY_ERR, 45$              ; 0762
                04  A7  DD 001AC        PUSHL   4(R7)                           ;
                    30  DD 001AF        PUSHL   #48                             ;
  00000000G 00      02  FB 001B1        CALLS   #2, FOR$$SIG_NO_LUB             ;
        0C          6E  E9 001B8 45$:   BLBC    V_KEY_VAL_ERR, 46$              ; 0764
                04  A7  DD 001BB        PUSHL   4(R7)                           ;
                    2D  DD 001BE        PUSHL   #45                             ;
  00000000G 00      02  FB 001C0        CALLS   #2, FOR$$SIG_NO_LUB             ;
                        04 001C7 46$:   RET                                     ; 0768

; Routine Size:  456 bytes,    Routine Base:  _FOR$CODE + 00D4


;  707          0769  1
```

```
;  709     0770  1 ROUTINE OPEN_ON_CONNECTED (                          ! Open on a connected unit
;  710     0771  1         OPEN,                                        ! Keyword vector
;  711     0772  1         L_UNWIND_ACTION                              ! Unwind action
;  712     0773  1     ) : CALL_CCB =
;  713     0774  1
;  714     0775  1 !++
;  715     0776  1 ! FUNCTIONAL DESCRIPTION:
;  716     0777  1 !
;  717     0778  1 !       This routine implements the FORTRAN-77 concept of open on
;  718     0779  1 !       a connected unit.
;  719     0780  1 !
;  720     0781  1 !       If an OPEN is done for a unit which is already open, one of two
;  721     0782  1 !       things happen:
;  722     0783  1 !           1.  If the filename specification in the OPEN is the same as
;  723     0784  1 !               the same as the file currently open, or if the filename
;  724     0785  1 !               is omitted but the unit is already open, then the value
;  725     0786  1 !               of BLANK= is set according to the keyword list.
;  726     0787  1 !           2.  If the filename specification in the OPEN is not the same
;  727     0788  1 !               as the file currently open, the old file is closed and
;  728     0789  1 !               the new one is opened.
;  729     0790  1 !
;  730     0791  1 ! FORMAL PARAMETERS:
;  731     0792  1 !
;  732     0793  1 !       OPEN.rl.ra              Sorted keyword list from OPEN
;  733     0794  1 !       L_UNWIND_ACTION.ml.r    Unwind action in case of an error
;  734     0795  1 !
;  735     0796  1 ! IMPLICIT INPUTS:
;  736     0797  1 !
;  737     0798  1 !       CCB                     Global I/O database register
;  738     0799  1 !
;  739     0800  1 ! IMPLICIT OUTPUTS:
;  740     0801  1 !
;  741     0802  1 !       LUB$V_NULLBLNK
;  742     0803  1 !
;  743     0804  1 ! ROUTINE VALUE:
;  744     0805  1 !
;  745     0806  1 !       True (1) if no further OPEN processing is needed (case 1)
;  746     0807  1 !       False (0) otherwise (case 2)
;  747     0808  1 !
;  748     0809  1 ! SIDE EFFECTS:
;  749     0810  1 !
;  750     0811  1 !       Possibly closes the currently open file
;  751     0812  1 !--
;  752     0813  1
;  753     0814  2     BEGIN
;  754     0815  2
;  755     0816  2     EXTERNAL REGISTER
;  756     0817  2         CCB : REF $FOR$CCB_DECL;
;  757     0818  2
;  758     0819  2     MAP
;  759     0820  2         OPEN : REF VECTOR [OPEN$K_KEY_MAX + 1];
;  760     0821  2
;  761     0822  2     LOCAL
;  762     0823  2         FAB : BLOCK [FAB$C_BLN, BYTE],           ! FAB block
;  763     0824  2         NAM : BLOCK [NAM$C_BLN, BYTE],           ! NAM block
;  764     0825  2         RES_NAME : VECTOR [NAM$C_MAXRSS, BYTE], ! Resultant name string
;  765     0826  2         RES_LEN,                                 ! Resultant string length
```

```
;  766     0827  2          DEF_NAME : VECTOR [10, BYTE],          ! Default name string
;  767     0828  2          NAM_DSC : REF DSC$DESCRIPTOR,          ! FILE/DEFAULTFILE descriptor
;  768     0829  2          UNIT,                                  ! Logical unit number
;  769     0830  2          RMS_STATUS;                            ! RMS condition status
;  770     0831
;  771     0832  1 !+
;  772     0833  1 ! Set up FAB and NAM blocks
;  773     0834  1 !-
;  774     0835  1     CH$FILL (0, FAB$C_BLN, FAB);
;  775     0836  1     CH$FILL (0, NAM$C_BLN, NAM);
;  776     0837  1     FAB [FAB$B_BID] = FAB$C_BID;
;  777     0838  1     FAB [FAB$B_BLN] = FAB$C_BLN;
;  778     0839  1     NAM [NAM$B_BID] = NAM$C_BID;
;  779     0840  1     NAM [NAM$B_BLN] = NAM$C_BLN;
;  780     0841  1     FAB [FAB$L_NAM] = NAM;
;  781     0842  1 !+
;  782     0843  1 ! Set up common default value for FILE and DEFAULTFILE if needed
;  783     0844  1 !-
;  784     0845  1     UNIT = .OPEN [OPEN$K_UNIT];
;  785     0846  1     IF .OPEN [OPEN$K_NAME] EQLA 0 OR
;  786     0847  1        .OPEN [OPEN$K_DEFAULTF] EQLA 0
;  787     0848  1     THEN
;  788     0849  1         BEGIN
;  789     0850  1         DEF_NAME [0] = %C'F';
;  790     0851  1         DEF_NAME [1] = %C'O';
;  791     0852  1         DEF_NAME [2] = %C'R';
;  792     0853  1         DEF_NAME [3] = ((.UNIT/100) MOD 10) + %C'0';
;  793     0854  1         DEF_NAME [4] = ((.UNIT/10) MOD 10) + %C'0';
;  794     0855  1         DEF_NAME [5] = ((.UNIT) MOD 10) + %C'0';
;  795     0856  1         DEF_NAME [6] = %C'.';
;  796     0857  1         DEF_NAME [7] = %C'D';
;  797     0858  1         DEF_NAME [8] = %C'A';
;  798     0859  1         DEF_NAME [9] = %C'T';
;  799     0860  1         END;
;  800     0861  1
;  801     0862  1 !+
;  802     0863  1 ! Set up DEFAULTFILE name
;  803     0864  1 !-
;  804     0865  1
;  805     0866  1     NAM_DSC = .OPEN [OPEN$K_DEFAULTF];
;  806     0867  1
;  807     0868  1     IF (.NAM_DSC NEQ 0)
;  808     0869  1     THEN
;  809     0870  1         BEGIN
;  810     0871  1 !+
;  811     0872  1 ! Default file name was specified.  Check for proper length then
;  812     0873  1 ! use it.
;  813     0874  1 !-
;  814     0875  1         IF ((.NAM_DSC [DSC$W_LENGTH] GTRU 255) OR (.NAM_DSC [DSC$W_LENGTH] EQL 0))
;  815     0876  1         THEN
;  816     0877  1             FOR$$SIG_NO_LUB (FOR$K_FILNAMSPE, .UNIT);
;  817     0878  1
;  818     0879  1         FAB [FAB$B_DNS] = .NAM_DSC [DSC$W_LENGTH];
;  819     0880  1         FAB [FAB$L_DNA] = .NAM_DSC [DSC$A_POINTER];
;  820     0881  1         END
;  821     0882  1     ELSE
;  822     0883  1         BEGIN
```

```
;  823     0884  3 !+
;  824     0885  3 ! DEFAULTFILE not specified, use name of FORnnn.DAT
;  825     0886  3 !-
;  826     0887  3         FAB [FAB$B_DNS] = %CHARCOUNT ('FORnnn.DAT');
;  827     0888  3         FAB [FAB$L_DNA] = DEF_NAME;
;  828     0889  2         END;
;  829     0890  2
;  830     0891  2 !+
;  831     0892  2 ! Set up file name
;  832     0893  2 !-
;  833     0894  2     NAM_DSC = .OPEN [OPEN$K_NAME];
;  834     0895  2
;  835     0896  3     IF (.NAM_DSC NEQ 0)
;  836     0897  2     THEN
;  837     0898  3         BEGIN
;  838     0899  3 !+
;  839     0900  3 ! File name was specified.  Check for proper length then
;  840     0901  3 ! use it.
;  841     0902  3 !-
;  842     0903  3
;  843     0904  4         IF ((.NAM_DSC [DSC$W_LENGTH] GTRU 255) OR (.NAM_DSC [DSC$W_LENGTH] EQL 0))
;  844     0905  3         THEN
;  845     0906  3             FOR$$SIG_NO_LUB (FOR$K_FILNAMSPE, .UNIT);
;  846     0907  3
;  847     0908  3         FAB [FAB$B_FNS] = .NAM_DSC [DSC$W_LENGTH];
;  848     0909  3         FAB [FAB$L_FNA] = .NAM_DSC [DSC$A_POINTER];
;  849     0910  3         END
;  850     0911  2     ELSE
;  851     0912  3         BEGIN
;  852     0913  3 !+
;  853     0914  3 ! File name not specified, use name of FORnnn which may be
;  854     0915  3 ! a logical name.
;  855     0916  3 !-
;  856     0917  3         FAB [FAB$B_FNS] = %CHARCOUNT ('FORnnn');
;  857     0918  3         FAB [FAB$L_FNA] = DEF_NAME;
;  858     0919  2         END;
;  859     0920  2
;  860     0921  2 !+
;  861     0922  2 ! Set up resultant name string
;  862     0923  2 !-
;  863     0924  2     NAM [NAM$B_ESS] = NAM [NAM$B_RSS] = NAM$C_MAXRSS;
;  864     0925  2     NAM [NAM$L_ESA] = NAM [NAM$L_RSA] = RES_NAME;
;  865     0926  2 !+
;  866     0927  2 ! Parse and search for the file to get the resultant name
;  867     0928  2 !-
;  868     0929  2     RMS_STATUS = $PARSE (FAB = FAB);
;  869     0930  2
;  870     0931  2     IF (.RMS_STATUS) THEN $SEARCH (FAB = FAB) ELSE FOR$$SIG_NO_LUB (FOR$K_FILNAMSPE, .UNIT, FAB);
;  871     0932  2
;  872     0933  2 !+
;  873     0934  2 ! Specifically forbid wildcards in file name.
;  874     0935  2 !-
;  875     0936  2
;  876     0937  3     IF (.NAM [NAM$V_WILDCARD])
;  877     0938  2     THEN
;  878     0939  3         BEGIN
;  879     0940  3         NAM [NAM$L_ESA] = 0;                    ! Invalidate result string
```

```
;  880     0941  }          NAM [NAM$L_RSA] = 0;
;  881     0942  }          FAB [FAB$L_STS] = 0;                        ! Invalidate statuses
;  882     0943  }          FAB [FAB$L_STV] = 0;
;  883     0944  }          FOR$$SIG_NO_LUB (FOR$K_FILNAMSPE, .UNIT, FAB);
;  884     0945  }          END;
;  885     0946
;  886     0947  !+
;  887     0948  ! See if the resultant name matches that stored in the LUB
;  888     0949  ! or if the name was not given and the unit is open.
;  889     0950  !-
;  890     0951  }      RES_LEN = MAX (.NAM [NAM$B_RSL], .NAM [NAM$B_ESL]);
;  891     0952  }
;  892     0953  }      IF ((CH$EQL (.RES_LEN, RES_NAME, .CCB [LUB$B_RSL], .CCB [LUB$A_RSN], %C' '))        !
;  893     0954  }          OR ((.OPEN [OPEN$K_NAME] EQL 0) AND .CCB [LUB$V_OPENED]))
;  894     0955  }      THEN
;  895     0956  }          BEGIN
;  896     0957  !+
;  897     0958  ! Names match, change BLANK= value only.
;  898     0959  !-
;  899     0960  }
;  900     0961  }          CASE .OPEN [OPEN$K_BLANK] FROM 0 TO OPEN$K_BLK_NUL OF
;  901     0962  }              SET
;  902     0963  }
;  903     0964  }              [0] :
;  904     0965  }              ;                                       ! Make no changes
;  905     0966  }
;  906     0967  }              [OPEN$K_BLK_ZER] :                      ! BLANK='ZERO'
;  907     0968  }                  CCB [LUB$V_NULLBLNK] = 0;
;  908     0969  }
;  909     0970  }              [OPEN$K_BLK_NUL] :                      ! BLANK='NULL'
;  910     0971  }                  CCB [LUB$V_NULLBLNK] = 1;
;  911     0972  }
;  912     0973  }              [OUTRANGE] :
;  913     0974  }                  FOR$$SIG_NO_LUB (FOR$K_INVARGFOR, .UNIT, FAB);
;  914     0975  }              TES;
;  915     0976  }
;  916     0977  !+
;  917     0978  ! BLANK= set, now pop the LUB/RAB/ISB and return to FOR$OPEN
;  918     0979  !-
;  919     0980  }          FOR$$CB_POP ();
;  920     0981  }          .L_UNWIND_ACTION = FOR$K_UNWINDNOP;
;  921     0982  }          RETURN 1;                                   ! No more OPEN processing needed
;  922     0983  }          END
;  923     0984  }      ELSE
;  924     0985  }          BEGIN
;  925     0986  !+
;  926     0987  ! File names do not match; close current file, open new one.
;  927     0988  !-
;  928     0989  }
;  929     0990  }          IF NOT FOR$$CLOSE_FILE () THEN FOR$$SIG_NO_LUB (FOR$K_CLOERR, .UNIT, FAB);
;  930     0991  }
;  931     0992  }          FOR$$CB_POP ();
;  932     0993  }          .L_UNWIND_ACTION = FOR$K_UNWINDNOP;
;  933     0994  !+
;  934     0995  ! Now, try to initiate re-opening of this unit
;  935     0996  !-
;  936     0997  }
```

```
;  937          0998  3          FOR$$CB_PUSH (.UNIT, LUB$K_LUN_MIN);
;  938          0999  3          .L_UNWIND_ACTION = FOR$K_UNWINDPOP;
;  939          1000  3
;  940          1001  4          IF ((.CCB [LUB$V_OPENED]) OR (.CCB [LUB$V_DEALLOC]))
;  941          1002  4          THEN
;  942          1003  5              FOR$$SIGNAL_STO (FOR$K_RECIO_OPE);
;  943          1004  4
;  944          1005  4          END;
;  945          1006  2
;  946          1007  2      RETURN 0;                                   ! Continue OPEN processing
;  947          1008  1      END;                                        ! of routine OPEN_ON_CONNECTED


                                                                        .EXTRN  SYS$PARSE, SYS$SEARCH

                                                        01FC 00000 OPEN_ON_CONNECTED:
                                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8                       ; 0770
                                                58 00000000G   00  9E 00002          MOVAB   FOR$$CB_POP, R8
                                                57 00000000G   00  9E 00009          MOVAB   FOR$$SIG_NO_LUB, R7
                                                5E      FE44   CE  9E 00010          MOVAB   -444(SP), SP
    0050    8F          00                      6E             00  2C 00015          MOVC5   #0, (SP), #0, #80, FAB          ; 0835
                                                       B0      AD       0001C
    0060    8F          00                      6E             00  2C 0001E          MOVC5   #0, (SP), #0, #96, NAM          ; 0836
                                                     FF50      CD       00025
                                          B0    AD      5003   8F  B0 00028          MOVW    #20483, FAB                     ; 0837
                                      FF50    CD      6002   8F  B0 0002E          MOVW    #24578, NAM                     ; 0839
                                        D8    AD      FF50   CD  9E 00035          MOVAB   NAM, FAB+40                     ; 0841
                                              54        04   AC  D0 0003B          MOVL    OPEN, R4                        ; 0845
                                              55        04   A4  D0 0003F          MOVL    4(R4), UNIT
                                                             56  D4 00043          CLRL    R6                              ; 0846
                                                        38   A4  D5 00045          TSTL    56(R4)
                                                             04  12 00048          BNEQ    1$
                                                             56  D6 0004A          INCL    R6
                                                             05  11 0004C          BRB     2$
                                                         68  A4  D5 0004E  1$:     TSTL    104(R4)                         ; 0847
                                                             4B  12 00051          BNEQ    3$
                                                    6E     4F46  8F  B0 00053  2$:     MOVW    #20294, DEF_NAME                ; 0850
                                                02  AE       52  8F  90 00058          MOVB    #82, DEF_NAME+2                 ; 0852
                                          52        55 00000064  8F  C7 0005D          DIVL3   #100, UNIT, R2                  ; 0853
                                7E        00        52          01  7A 00065          EMUL    #1, R2, #0, -(SP)
                                52        52        8E          0A  7B 0006A          EDIV    #10, (SP)+, R2, R2
                                      03  AE        52          30  81 0006F          ADDB3   #48, R2, DEF_NAME+3
                                          52        55          0A  C7 00074          DIVL3   #10, UNIT, R2                   ; 0854
                                7E        00        52          01  7A 00078          EMUL    #1, R2, #0, -(SP)
                                52        52        8E          0A  7B 0007D          EDIV    #10, (SP)+, R2, R2
                                      04  AE        52          30  81 00082          ADDB3   #48, R2, DEF_NAME+4
                                7E        00        55          01  7A 00087          EMUL    #1, UNIT, #0, -(SP)             ; 0855
                                50        50        8E          0A  7B 0008C          EDIV    #10, (SP)+, R0, R0
                                      05  AE        50          30  81 00091          ADDB3   #48, R0, DEF_NAME+5
                                    06    AE 5441442E  8F  D0 00096          MOVL    #1413563438, DEF_NAME+6                 ; 0856
                                              52        68  A4  D0 0009E  3$:     MOVL    104(R4), NAM_DSC                ; 0866
                                                             1D  13 000A2          BEQL    6$                              ; 0868
                                  00FF    8F                62  B1 000A4          CMPW    (NAM_DSC), #255                 ; 0875
                                                             04  1A 000A9          BGTRU   4$
                                                             62  B5 000AB          TSTW    (NAM_DSC)
                                                             07  12 000AD          BNEQ    5$
```

```
                                        55  DD 000AF 4$:    PUSHL   UNIT                                    ; 0877
                                        2B  DD 000B1        PUSHL   #43
                                    67  02  FB 000B3        CALLS   #2, FOR$$SIG_NO_LUB
                                E5  AD  62  90 000B6 5$:    MOVB    (NAM_DSC), FAB+53                       ; 0879
                                E0  AD  04  A2 DO 000BA     MOVL    4(NAM_DSC), FAB+48                      ; 0880
                                        08  11 000BF        BRB     7$                                      ; 0868
                                E5  AD      0A  90 000C1 6$:    MOVB    #10, FAB+53                         ; 0887
                                E0  AD      6E  9E 000C5    MOVAB   DEF_NAME, FAB+48                        ; 0888
                                    52  38  A4  DO 000C9 7$:    MOVL    56(R4), NAM_DSC                     ; 0894
                                        1D  13 000CD        BEQL    10$                                     ; 0896
                            00FF    8F      62  B1 000CF    CMPW    (NAM_DSC), #255                         ; 0904
                                        04  1A 000D4        BGTRU   8$
                                        62  B5 000D6        TSTW    (NAM_DSC)
                                        07  12 000D8        BNEQ    9$
                                        55  DD 000DA 8$:    PUSHL   UNIT                                    ; 0906
                                        2B  DD 000DC        PUSHL   #43
                                    67  02  FB 000DE        CALLS   #2, FOR$$SIG_NO_LUB
                                E4  AD  62  90 000E1 9$:    MOVB    (NAM_DSC), FAB+52                       ; 0908
                                DC  AD  04  A2 DO 000E5     MOVL    4(NAM_DSC), FAB+44                      ; 0909
                                        08  11 000EA        BRB     11$                                     ; 0896
                                E4  AD      06  90 000EC 10$:   MOVB    #6, FAB+52                          ; 0917
                                DC  AD      6E  9E 000F0    MOVAB   DEF_NAME, FAB+44                        ; 0918
                            FF52    CD      01  8E 000F4 11$:   MNEGB   #1, NAM+2                           ; 0924
                            FF5A    CD      01  8E 000F9    MNEGB   #1, NAM+10
                                50  0C  AE  9E 000FE        MOVAB   RES_NAME, R0                            ; 0925
                            FF54    CD      50  D0 00102    MOVL    R0, NAM+4
                            FF5C    CD      50  D0 00107    MOVL    R0, NAM+12
                                    B0  AD  9F 0010C        PUSHAB  FAB                                     ; 0929
                    00000000G       00      01  FB 0010F    CALLS   #1, SYS$PARSE
                                    0C      50  E9 00116    BLBC    RMS_STATUS, 12$                         ; 0931
                                    B0  AD  9F 00119        PUSHAB  FAB
                    00000000G       00      01  FB 0011C    CALLS   #1, SYS$SEARCH
                                            0A  11 00123    BRB     13$
                                    B0  AD  9F 00125 12$:   PUSHAB  FAB
                                        55  DD 00128        PUSHL   UNIT
                                        2B  DD 0012A        PUSHL   #43
                                67      03  FB 0012C        CALLS   #3, FOR$$SIG_NO_LUB
                                15  85  AD  E9 0012F 13$:   BLBC    NAM+53, 14$                             ; 0937
                                FF5C    CD  D4 00133        CLRL    NAM+12                                  ; 0940
                                FF54    CD  D4 00137        CLRL    NAM+4                                   ; 0941
                                    B8  AD  7C 0013B        CLRQ    FAB+8                                   ; 0942
                                    B0  AD  9F 0013E        PUSHAB  FAB                                     ; 0944
                                        55  DD 00141        PUSHL   UNIT
                                        2B  DD 00143        PUSHL   #43
                                67      03  FB 00145        CALLS   #3, FOR$$SIG_NO_LUB
                                50  FF53    CD  9A 00148 14$:   MOVZBL  NAM+3, R0                           ; 0951
                                50  FF5B    CD  91 0014D    CMPB    NAM+11, R0
                                        05  1B 00152        BLEQU   15$
                                50  FF5B    CD  9A 00154    MOVZBL  NAM+11, R0
                                51          50  D0 00159 15$:   MOVL    R0, RES_LEN
                                50      F7  AB  9A 0015C    MOVZBL  -9(CCB), R0                             ; 0953
50          20      0C  AE                  51  2D 00160    CMPC5   RES_LEN, RES_NAME, #32, R0, @-8(CCB)
                                    F8  BB      00166
                                        07  13 00168        BEQL    16$
                                31          56  E9 0016A    BLBC    R6, 21$                                 ; 0954
                                2D      FC  AB  E9 0016D    BLBC    -4(CCB), 21$
                    02          00      60  A4  CF 00171 16$:   CASEL   96(R4), #0, #2                       ; 0961
```

```
                0019          0012          001E    00176 17$:    .WORD   20$-17$,-
                                                                  18$-17$,-
                                                                  19$-17$
                                      B0  AD  9F 0017C          PUSHAB  FAB                             ; 0974
                                          DD 0017F              PUSHL   UNIT
                                          DD 00181              PUSHL   #48
                                    67    03  FB 00183          CALLS   #3, FOR$$SIG_NO_LUB
                                          0C  11 00186          BRB     20$
                              FF    AB    40  8F  8A 00188 18$: BICB2   #64, -1(CCB)                 ; 0968
                                          05  11 0018D          BRB     20$
                              FF    AB    40  8F  8B 0018F 19$: BISB2   #64, -1(CCB)                 ; 0971
                                          68  16 00194 20$:     JSB     FOR$$CB_POP                   ; 0980
                              08    BC        01  D0 00196      MOVL    #1, @L_ONWIND_ACTION          ; 0981
                                    50        01  D0 0019A      MOVL    #1, R0                        ; 0982
                                              04  04 0019D      RET
                    00000000G 00              00  FB 0019E 21$: CALLS   #0, FOR$$CLOSE_FILE           ; 0990
                                    0A        50  E8 001A5      BLBS    R0, 22$
                                      B0      AD  9F 001A8      PUSHAB  FAB
                                              55  DD 001AB      PUSHL   UNIT
                                              1C  DD 001AD      PUSHL   #28
                                    67        03  FB 001AF      CALLS   #3, FOR$$SIG_NO_LUB
                                              68  16 001B2 22$: JSB     FOR$$CB_POP                   ; 0992
                              08    BC        01  D0 001B4      MOVL    #1, @L_ONWIND_ACTION          ; 0993
                                              50  D4 001B8      CLRL    R0                            ; 0998
                                    52        55  D0 001BA      MOVL    UNIT, R2
                                        00000000G 00  16 001BD  JSB     FOR$$CB_PUSH
                                            08    BC  D4 001C3  CLRL    @L_UNWIND_ACTION              ; 0999
                                    05      FC    AB  E8 001C6  BLBS    -4(CCB), 23$                  ; 1001
                09        FF    AB              04  E1 001CA  BBC     #4, -1(CCB), 24$
                                              28  DD 001CF 23$: PUSHL   #40                           ; 1003
                    00000000G 00              01  FB 001D1      CALLS   #1, FOR$$SIGNAL_STO
                                              50  D4 001D8 24$: CLRL    R0                            ; 1007
                                                  04 001DA      RET                                   ; 1008

; Routine Size:  475 bytes,    Routine Base:  _FOR$CODE + 029C

;  948          1009  1 END                                    ! End of FOR$OPEN module
;  949          1010  1
;  950          1011  0 ELUDOM



;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _FOR$CODE                       1143  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                              Library Statistics
;
```

```
;                                  --------  Symbols  --------      Pages      Processing
;       File                       Total     Loaded   Percent     Mapped     Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1    9776       32         0        581       00:01.1
;  _$255$DUA28:[FORRTL.OBJ]FORLIB.L32;1  711      223        31         52       00:00.5
;  _$255$DUA28:[FORRTL.OBJ]RTLLIB.L32;1   36        0         0          8       00:00.1



;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:FOROPEN/OBJ=OBJ$:FOROPEN MSRC$:FOROPEN/UPDATE=(ENH$:FOROPEN)

; Size:         1143 code + 0 data bytes
; Run Time:       00:25.4
; Elapsed Time:   01:10.9
; Lines/CPU Min:   2392
; Lexemes/CPU-Min: 15555
; Memory Used:  233 pages
; Compilation Complete
```

FORRAB
LIS

FORREADDF
LIS

FOROPNKEY
LIS

FORPAUSE
LIS

FORRANDOM
LIS

FOROPEN
LIS

FOROPENDE
LIS

FORREADDO
LIS